Part-Object Relational Visual Saliency

Liu, Yi; Zhang, Dingwen; Zhang, Qiang; Han, Jungong

Published in:
IEEE Transactions on Pattern Analysis and Machine Intelligence

DOI:
10.1109/TPAMI.2021.3053577

Publication date:
2022

Citation for published version (APA):
Liu, Y., Zhang, D., Zhang, Q., & Han, J. (2022). Part-Object Relational Visual Saliency. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *44*(7), 3688-3704. [9334445].
https://doi.org/10.1109/TPAMI.2021.3053577

Part-Object Relational Visual Saliency

Yi Liu, Dingwen Zhang, Qiang Zhang, and Jungong Han

Abstract—Recent years have witnessed a big leap in automatic visual saliency detection attributed to advances in deep learning, especially Convolutional Neural Networks (CNNs). However, inferring the saliency of each image part separately, as was adopted by most CNNs methods, inevitably leads to an incomplete segmentation of the salient object. In this paper, we describe how to use the property of part-object relations endowed by the Capsule Network (CapsNet) to solve the problems that fundamentally hinge on relational inference for visual saliency detection. Concretely, we put in place a two-stream strategy, termed Two-Stream Part-Object RelaTional Network (TSPORTNet), to implement CapsNet, aiming to reduce both the network complexity and the possible redundancy during capsule routing. Additionally, taking into account the correlations of capsule types from the preceding training images, a correlation-aware capsule routing algorithm is developed for more accurate capsule assignments at the training stage, which also speeds up the training dramatically. By exploring part-object relationships, TSPORTNet produces a capsule wholeness map, which in turn aids multi-level features in generating the final saliency map. Experimental results on five widely-used benchmarks show that our framework consistently achieves state-of-the-art performance. The code can be found on https://github.com/liuyi1989/TSPORTNet.

Index Terms—Salient object detection, capsule network, part-object relationships.

1 Introduction

The task of salient object detection is to mimic the human ability to identify the visually distinctive objects or regions in a scene and then to segment them out from the background. Serving as a preprocessing step, salient object detection has been prevalent in a variety of computer vision and cognitive science applications, including image segmentation [1, 2], image fusion [3], object recognition [4, 5], image and video compression [6–8], image retrieval [9, 10], and person re-identification [11].

Most early salient object detectors [12–16] advocate the usage of hand-crafted features to capture the contrast information in an image. However, those features become powerless when grabbing the visual difference in complex scenes, thus leading to a performance bottleneck. The emergence of Convolutional Neural Networks (CNNs) makes it possible to capture more primitive scene semantics, thereby boosting the development of salient object detection [17–20].

Most salient object detectors based on CNNs learn deep features via abundant kernels on each image region, intending to capture the contrast of different image parts. The entire saliency map is made up by assembling those of individual parts. However, this mechanism has some latent issues remaining under cover. First, compared with the background, salient parts that are vested with low contrast may be easily mislabeled as non-salient, as evidenced by the experimental results in the top row of Fig. 1. Secondly, those locally low-contrast parts inside a salient object will be likely missed in the final saliency map, as can be observed in the bottom row of Fig. 1. Altogether, it is not surprising that most existing CNN based approaches end up with an incomplete segmentation of the salient object.

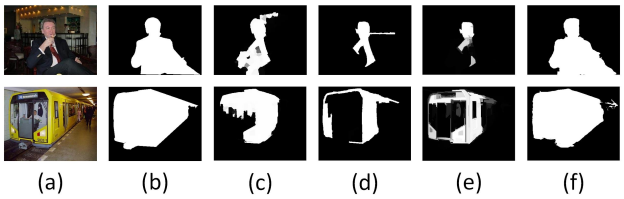


Fig. 1. Some examples showing the incomplete segmentation of the salient object. (a) Input image; (b) GT; (c) DLS [21]; (d) ELE [22]; (e) MDF [17]; (f) OURS.

In other words, there exist some intrinsic relationships between an object and its parts, which have long been overlooked by existing CNN based salient object detection algorithms. A salient object is usually composed of several associated parts. From another perspective, these relevant parts, which share familiar properties of the object, have great potential to make up a complete object in an image. This reveals the natural relationships between object parts and the complete object, *i.e.*, those parts familiar to an object will be clustered together to form a complete object in a full image. A graphical explanation can be found in Fig. 2. Inspired by these observations, we introduce the property of part-object relationships for salient object detection in this paper, which provides a solution to the problem of incomplete salient object segmentation.

Recently, a new deep learning architecture named Capsule Network (CapsNet) [23–25] has shown promising results for digits recognition and image classification. A capsule encapsulates a group of neurons whose outputs represent different properties of an entity, *e.g.*, an object or an object part. Contrary to CNNs, whose primary role is to learn the contrast cues per image region, CapsNet assigns associated parts (low-level capsules) to their whole object (high-level capsule), which can be graphically illustrated

- *Yi Liu, Dingwen Zhang, and Qiang Zhang are with School of Mechano-Electronic Engineering, Xidian University, China. E-mail: liuyi0089@gmail.com, zdw@xidian.edu.cn, qzhang@xidian.edu.cn.*
- *Jungong Han is with Department of Computer Science, Aberystwyth University, U.K. E-mail: jungonghan77@gmail.com.*
- *Equally corresponding authors: Qiang Zhang and Jungong Han.*

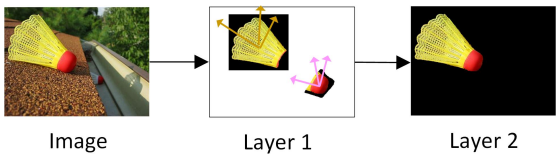


Fig. 2. Graphical illustration of the part-object relationships. Coordinates with different colors represent poses of different parts. Given an image, an object (badminton) is composed by several associated parts (base and wing). Conversely, these relevant parts (base and wing), which share familiar properties of the object (badminton), can make up a complete object (badminton). To this end, CapsNet first infers poses of different parts of the image, and then votes associated parts to their whole object.

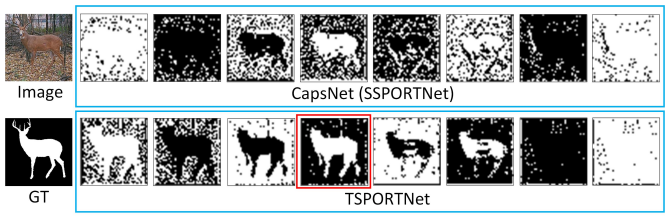


Fig. 3. Comparison between TSPORTNet and CapsNet (*i.e.*, Single-Stream PORTNet (SSPORTNet)). All types of capsules of the second convolutional capsule layer in TSPORTNet and CapsNet are displayed here. Compared with the original CapsNet, the proposed TSPORTNet produces more discriminative capsules, which are capable of identifying the salient object from backgrounds.

in Fig. 2. The entire framework is accomplished by finding similar votes from parts to the whole object layer by layer.

However, a primitive adaptation of CapsNet to salient object detection may not work, which can be explained from three aspects. First, each low-level capsule essentially relates to a subset but not full set types of high-level capsules. That is to say, the current capsule routing scheme, which enforces each low-level capsule to vote for all types of high-level ones, is at the risk of generating fake assignments. This is well verified by non-distinguishable capsules produced by CapsNet in Fig. 3, which eventually causes the failure to identify the salient object. Secondly, despite the great performance, CapsNet is known for its high computational demand even when carrying out simple tasks, like digital image classification. Therefore, using CapsNet to deal with large-scale dense prediction tasks (*e.g.*, salient object detection), which is in theory much more complicated than the task of image classification, without simplifying key algorithmic components is impractical. Thirdly, unlike image classification, the pixel-level salient object detection task is confronted with more complex scenes, where complicated part-object relationships will challenge the current routing algorithm.

To mitigate the above issues, we present, in this paper, a deep Two-Stream Part-Object RelaTional Network (TSPORTNet) for salient object detection, where a new two-stream strategy is adopted to implement CapsNet for sake of better exploring part-object relationships. Specifically, those primary capsules are divided into two groups, each being fed into two streams, respectively. Given a stream, each capsule is voted for one group types of capsules via a locally-connected routing.[1] As shown in Fig. 3, those capsules generated by the proposed TSPORTNet, especially the one marked by the red box, are distinctive enough to identify the salient object in a scene. These two streams are integrated through a fully-connected routing such that the relevant parts can be clustered to form a salient object. Compared to the original CapsNet, the network based on this two-stream strategy has much less to-be-trained parameters, thus making the training of TSPORTNet fairly efficient. Meanwhile, since we effectively simplify the capsule connections, the likelihood of having noisy assignments is reduced.

Besides, it is observed from Fig. 4 that different types of capsules represent different patterns, which might be changed if the input image changes. However, those types of capsules, which activate the salient object, remain unchanged, *e.g.*, type-4 of stream 1 and type-4 of stream 2 in Fig. 4. It indicates that pattern assignment, *i.e.*, routing one type of low-level capsules to one type of high-level capsules, does not necessarily change, as evidenced by the fact that strongly correlated types of capsules across adjacent layers are likely assigned together. The above observation inspires us to develop a more accurate and efficient capsule routing algorithm to conduct the capsule assignments task when dealing with complicated scenes. Specifically, we design a correlation-aware capsule routing algorithm at the training phase, where the correlations/similarities between different types of capsules across adjacent layers, computed from the previous images, are used to predict the capsule assignments for the next image. In doing so, on the one hand, the network training procedure will be accelerated by routing highly correlated capsules. On the other hand, robust capsule assignments can be achieved to explore the right part-object relationships in different scenes.

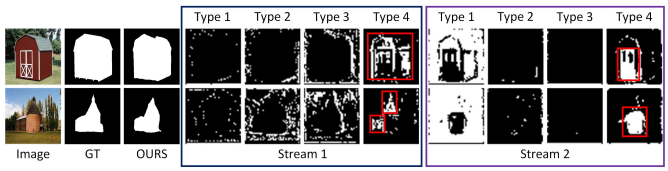


Fig. 4. Output capsules of two streams. Obviously, the pattern of each capsule type keeps up with the change of the input image. However, those types of capsules that activate the salient parts remain unchanged, *e.g.*, type-4 of stream 1 and type-4 of stream 2, which indicates that pattern assignments do not necessarily change, given different input images.

By exploring part-object relationships, TSPORTNet can detect the whole salient object and thus produce a Capsule Wholeness Map (CWM). Several CWMs are exhibited in Fig. 5. However, CWM so far does not look perfect due to: i) Blurriness around the object boundary; ii) The background being mixed with a lot of bright spots; iii) The salient region seeming unsmooth. To remedy these situations, we further supply CWM with abundant spatial details. Concretely, CWM acts as a guidance map for multi-level features

1. In our experiments, given the fixed total number of capsule types, we find that non-convergence occurs in the proposed model with 4 and 8 streams, each of which has too few types of capsules. However, the model works well with 2 streams provided. This indicates that it reaches a good trade-off between the type number of familiar high-level capsules and the type number of low-level capsules, given 2 streams.

extraction, thus resulting in a Wholeness Guidance network (WGNet). On the one hand, multi-level features will learn more accurate saliency cues under the guidance of CWM. On the other hand, spatial details will be provided to the CWM, thereby helping to generate a smoother saliency map.

Fig. 5. Output CWMs of TSPORTNet. Top: Images; Bottom: CWM. Some issues occur on CWM: blurry object boundaries, bright spots on the background region, and un-smooth salient regions.

Fig. 6 shows the architecture of our proposed salient object detection network. A preliminary version of our work was published in [26]. This paper reinforces the work in [26] with further architecture optimization and in-depth analysis of key components. The main extensions of this paper are twofold. First, we design a correlation-aware routing algorithm, on top of the further analysis about the existing capsule routing algorithm, to achieve more accurate capsule assignments. Secondly, instead of restoring the spatial details by using deconvolutional operations, we develop a more sophisticated network, namely WGNet, which allows multi-level features to provide spatial details for CWM. This way will obtain a smoother saliency map.

Our contributions are summarized as follows:

(1) We involve a new property, *i.e.*, part-object relationships, in salient object detection, which is implemented by CapsNet. To the best of our knowledge, this is the first attempt to apply CapsNet to explore part-object relationships for salient object detection.

(2) We propose a deep TSPORTNet for salient object detection, in which a new two-stream strategy is adopted to improve the CapsNet. The intention is to narrow down the searching space when the routing from a low-level capsule to the high-level capsules takes place. Doing so reduces the complexity of CapsNet significantly while lowering the likelihood of having noisy assignments.

(3) We design a correlation-aware capsule routing algorithm, which preserves capsule correlations between different capsule types across adjacent layers after every training image to make predictions for capsules routing in the following image. Such routing mechanism is beneficial to accelerate the network training procedure and learn more accurate capsule assignments as well as more primitive part-object relationships in a scene.

(4) We build a WGNet, in which the CWM output by TSPORTNet can guide multi-level features to restore spatial details for a smoother saliency map.

The remainder of this paper is organized as follows. Section 2 outlines an overview of CNN based salient object detection techniques and a brief introduction of CapsNet. Section 3 details the proposed deep salient object detection network. Section 4 provides some insight analysis of the proposed framework. Section 5 presents experiments evaluating the proposed model. Section 7 concludes this paper.

2 Related Work

Over the past two decades, a vast number of salient object detection methods have emerged. Traditional methods [13, 27–38] detect the salient object based on hand-crafted features. The descriptions of these methods are beyond the scope of this paper. Readers can gain a comprehensive understanding of these methods from [16]. Here, we focus on those salient object detection methods based on CNNs, which are most relevant to our work. Some works about CapsNet are also reviewed in this section.

2.1 CNNs Based Salient Object Detection

The development of CNNs has achieved substantial improvements in salient object detection. Zhao *et al.* [39] detected the salient object by jointly taking into account global context and local context under a unified deep learning framework. Li *et al.* [17] used CNNs to learn multi-scale deep features for saliency detection. Wang *et al.* [40] trained two subnetworks for local estimation and global search, respectively. These methods implemented saliency detection via patch-by-patch scanning, thus requiring large computational resources. To address this issue, saliency detection was conducted via a fully convolutional network. For example, Liu *et al.* [18] proposed an end-to-end deep hierarchical saliency detection framework, in which a coarsely global prediction was achieved by learning various global saliency cues, and then a hierarchical recurrent CNN was applied to refine the coarse prediction by making up the discarded details. Zhang *et al.* [41] proposed a multi-level feature aggregation network for salient object detection by integrating multi-level features into multiple resolutions, which well incorporated low-level fine details and high-level semantic knowledge. Liu *et al.* [42] learned to generate a contextual attention for each pixel, which was formulated by incorporating the global and local context. Zhang *et al.* [43] designed a gated bi-directional message passing module to integrate multi-level features in both shallow-to-deep and deep-to-shallow directions, which efficiently explored each level of features to detect salient objects. Zeng *et al.* [44] incorporated global semantics and high-resolution details for large-scale image saliency detection. Liu *et al.* [45] investigated the role played by the pooling layer in a real-time salient object detection framework.

Generally, the above methods infer the saliency through looking at the contrast information provided by learned deep features. While our method detects the salient object from a new perspective - by exploring part-object relationships, our framework assigns associated parts to a complete salient object. This can well solve the problem of incomplete segmentation of the salient object from backgrounds encountered by existing CNNs based methods.

2.2 CapsNet

Recently, a new neural network structure named Capsule Network (CapsNet), was developed by Hinton *et al.* [23]. A capsule is essentially a group of neurons, which are used to represent the instantiation parameters of a specific type of the entity such as pose (position, size, and orientation), deformation, texture, *etc*. In spite of its advances,

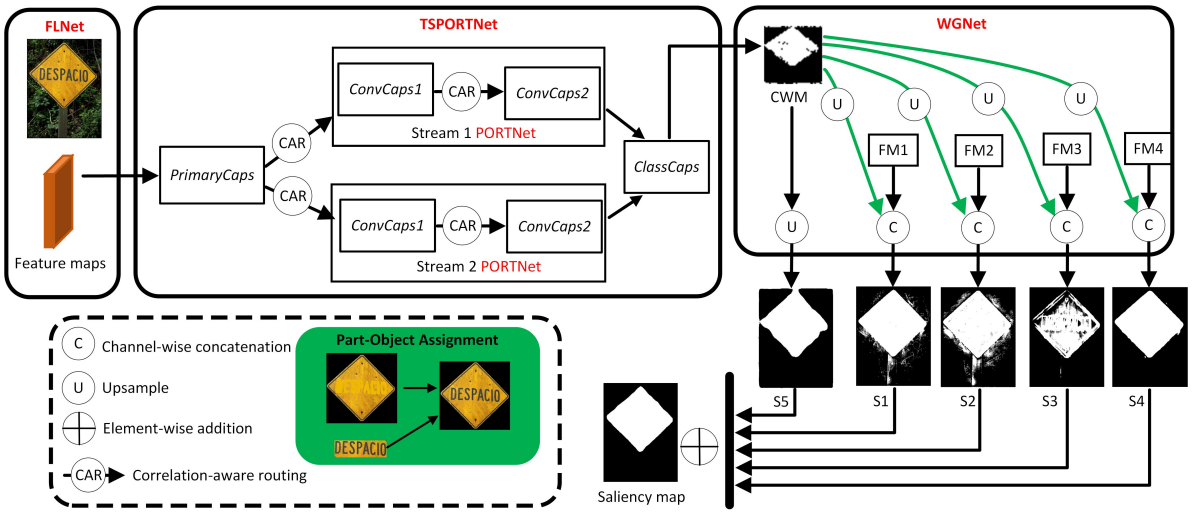


Fig. 6. Overview of the proposed network architecture. The image is first input to FLNet to learn deep features (described in the following Fig. 7), which are then fed to TSPORTNet. In TSPORTNet, capsules in the *PrimaryCaps* layer, constructed from deep feature maps, are divided into two groups, which are then fed into two identical streams. These two streams are integrated into the *ClassCaps* layer. By exploring part-object relationships, TSPORTNet produces a CWM, which is fed into WGNet to guide multi-level feature maps of FLNet (FMi ($i = 1, 2, 3, 4$) are described in Fig. 7). The final saliency map is computed by integrating the outputs of TSPORTNet and WGNet.

the CapsNet did not get much attention until Sabour *et al*. [24] implemented the vector CapsNet, in which a capsule encapsulated a group of neurons as a vector. An iterative dynamic routing algorithm was proposed to assign low-level capsules to the familiar high-level capsules via transformation weights, which learned to encode the intrinsic spatial relationship between a part and the whole as well as viewpoint invariant knowledge. One year later, Hinton *et al*. [25] consolidated their work by proposing a matrix CapsNet, in which each capsule contained a pose matrix and an activation probability. An iterative Expectation-Maximization (EM) algorithm was proposed to assign low-level capsules to high-level capsules or parts to wholes by finding tight clusters of high-dimensional votes that agreed in a mist of irrelevant votes. Compared with the vector CapsNet, the matrix CapsNet has two advantages: 1) The transformation matrix for the pose matrix adopted in the matrix CapsNet has much less parameters than that for the pose vector in the vector CapsNet. 2) The vector CapsNet uses the cosine similarity between two pose vectors to measure their agreement, while the matrix CapsNet adopts an iterative EM algorithm, which is proven to be better.

In view of its advances, some attempts have been made to apply CapsNet for several computer vision tasks. For instance, Zhao *et al.* [46] investigated the possibility of using CapsNet for text classification, in which the dynamic routing process was stabilized to alleviate the disturbance of some noisy capsules that might contain "background" information such as stop words and the words unrelated to specific categories. Duarte *et al.* [47] proposed a 3D VideoCapsuleNet, which could jointly perform pixel-wise action segmentation alongside action classification.

In this paper, we introduce the property of part-object relationships, which is explored by CapsNet, for salient object detection. Instead of directly adopting CapsNet, we enhance it with a two-stream strategy and a correlation-aware routing algorithm, thus making the new CapsNet specifically suitable for salient object detection.

2.3 View-based Representations

View-based representations have been studied for various problems. Biederman [48] proposed a recognition-by-component theory for image understanding, which pointed out the stable three-dimensional mental representations of objects were formed by manipulating a few simple geometric shapes. In [49], the authors investigated the evidence-based classification approach, which was modeled based on human perceived attributes and how such attribute states "evidence" for different classes. Solina and Bajcsy [50] sought to recover superquadrics from pre-segmented range images. [51] made superquadrics recovery from images more widespread. Krivic and Solina [52] carried out research on the possible use of part-level descriptions obtained by the Segmentor system [53], which was an object-based segmentation paradigm using superquadrics, for articulated objects recognition. Pentland [54] segmented an image into roughly convex component parts and extracted 3D deformable models for recognition and prediction. Felzenszwalb [55] used the deformable part models for cascade object detection. Girshick [56] expressed the deformable part model as an equivalent CNN by using distance transform pooling, object geometry filters, and maxout units. Zhao *et al.* [57] modeled a part ordinal relationship to improve the semantic object part segmentation. Wang *et al.* [58] employed object-level context for part segmentation guidance, and detailed part-level localization for object segmentation refinement.

Different from the previous methods that mostly represent each component by only spatial features (only neurons representing the existence probability of the component), in our model, the view representation for each component is represented by a capsule (*i.e.*, a group of neurons), which includes a pose matrix indicating the pose attributes of the component, and an activation value demonstrating the existence probability of the component. On top of that, we

segment out a complete object in a scene by detecting its associated parts based on their pose and activation representations, which is implemented by the capsule routing algorithm.

3 Proposed Method

Fig. 6 shows the overall architecture of the proposed deep salient object detection network. Our system begins with a Feature Learning Network (FLNet), in which the input image is represented by meaningful primitive features. Afterwards, these features are fed into the proposed Two-Stream Part-Object RelaTional Network (TSPORTNet). In TSPORTNet, those primary capsules in *PrimaryCaps*, constructed by deep feature maps, are divided into two groups, each being fed into two identical streams, namely PORTNet. These two streams are integrated by a fully-connected routing in *ClassCaps*. Thanks to the explored part-object relationships, TSPORTNet produces a Capsule Wholeness Map (CWM), which is further input into WGNet to guide multi-level features for the sake of smoothing the saliency map. The final saliency map is obtained by integrating those outputs of TSPORTNet and WGNet.

3.1 FLNet

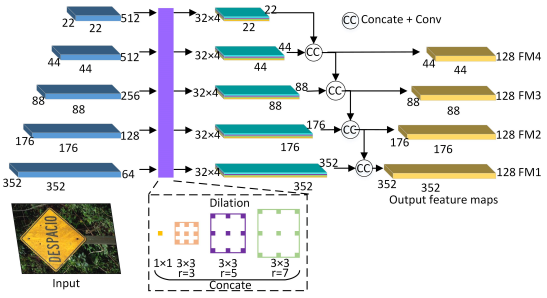


Fig. 7. The architecture of FLNet. The input image first undergoes five stacked convolutional layers, each of which is followed by concatenating four dilation convolutional layers. On top of that, different layers are integrated in a deep-to-shallow manner.

FLNet is designed for the purpose of learning rich features of the input image. The details of FLNet are displayed in Fig. 7. As observed in Fig. 7, the input image first undergoes five stacked convolutional layers. On top of that, four dilation convolutional layers [59] are stacked up at each stage, which have the same convolutional kernel size of 3×3 with different dilation rates, including 1, 3, 5, and 7. Such a structure helps to capture rich context information under various receptive fields without increasing the kernel scales. Besides, considering the fact that low-level feature maps aid in capturing fine details while high-level feature maps can grab semantic knowledge, deeper-level feature maps are integrated with shallower-level ones layer by layer until the shallowest stage, which efficiently aggregates multi-stage context information.

3.2 TSPORTNet

TSPORTNet aims to explore the part-object relationships in a scene, consisting of three stages, *i.e.*, capsules construction, two-stream PORTNet, and capsule classification. The details of TSPORTNet will be illustrated as follows.

3.2.1 Capsules Construction

Capsules are constructed based on the feature maps learned by FLNet, which is implemented by a Primary Capsule (*PrimaryCaps*) layer. Each capsule consists of a pose matrix with the dimension of 4×4 and an activation value, which represent the pose characteristics (such as an object part and an object) and the existence probability of the entity, respectively. Therefore, *PrimaryCaps* is formed by two branches including pose matrix construction and activation construction, which are aggregated for the capsules construction. Fig. 8 shows details of *PrimaryCaps*, which will be elaborated in the following.

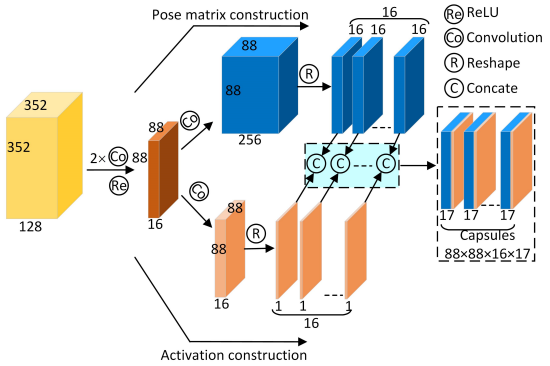


Fig. 8. Capsules construction. The context features of FLNet are downsampled for efficient computation. In the following, two branches emerge for pose matrix construction and activation construction, which are finally integrated to compose capsules.

The context features of FLNet (*i.e.*, $352 \times 352 \times 128$) are downsampled with two convolutions (stride: 2) layers and one ReLU layer to $88 \times 88 \times 16$ for efficient computation.

Pose matrix construction: The 16-channel feature maps ($88 \times 88 \times 16$) are first transformed to 256-channel feature maps ($88 \times 88 \times 256$) via one convolutional layer, which are then reshaped into $88 \times 88 \times 16 \times 16$ as the vectorized pose matrices[2] of 16 types of capsules.

Activation construction: The 16-channel feature maps ($88 \times 88 \times 16$) are first transformed to 16-channel feature maps ($88 \times 88 \times 16$) via one convolutional layer, which are reshaped into $88 \times 88 \times 16 \times 1$ as the activation information of 16 types of capsules.

Capsules construction: The vectorized pose matrices and activations are concatenated together to construct 16 types of capsules, *i.e.*, $88 \times 88 \times 16 \times 17$[3].

3.2.2 Two-Stream PORTNet

Those capsules obtained by *PrimaryCaps* are divided into two groups, each of which contains 8 types of capsules ($88 \times 88 \times 8 \times 17$). These two groups of capsules are fed into two identical streams, respectively. Each stream consists of two Convolutional Capsule (*ConvCaps1* and *ConvCaps2*) layers with 8 and 4 types of capsules, respectively. *ConvCaps1* and *ConvCaps2* share the same architecture, *i.e.*, enrich capsule types, compute capsules votes, and route capsules,

2. Here, the pose matrix of each capsule is lengthened as a vector for efficient storage.

3. Channel-(1 & 2) are the number of capsules of one type. In other words, each pixel position possesses a capsule. Channel-(3 & 4) are the number of capsule types and the one-capsule dimension, respectively.

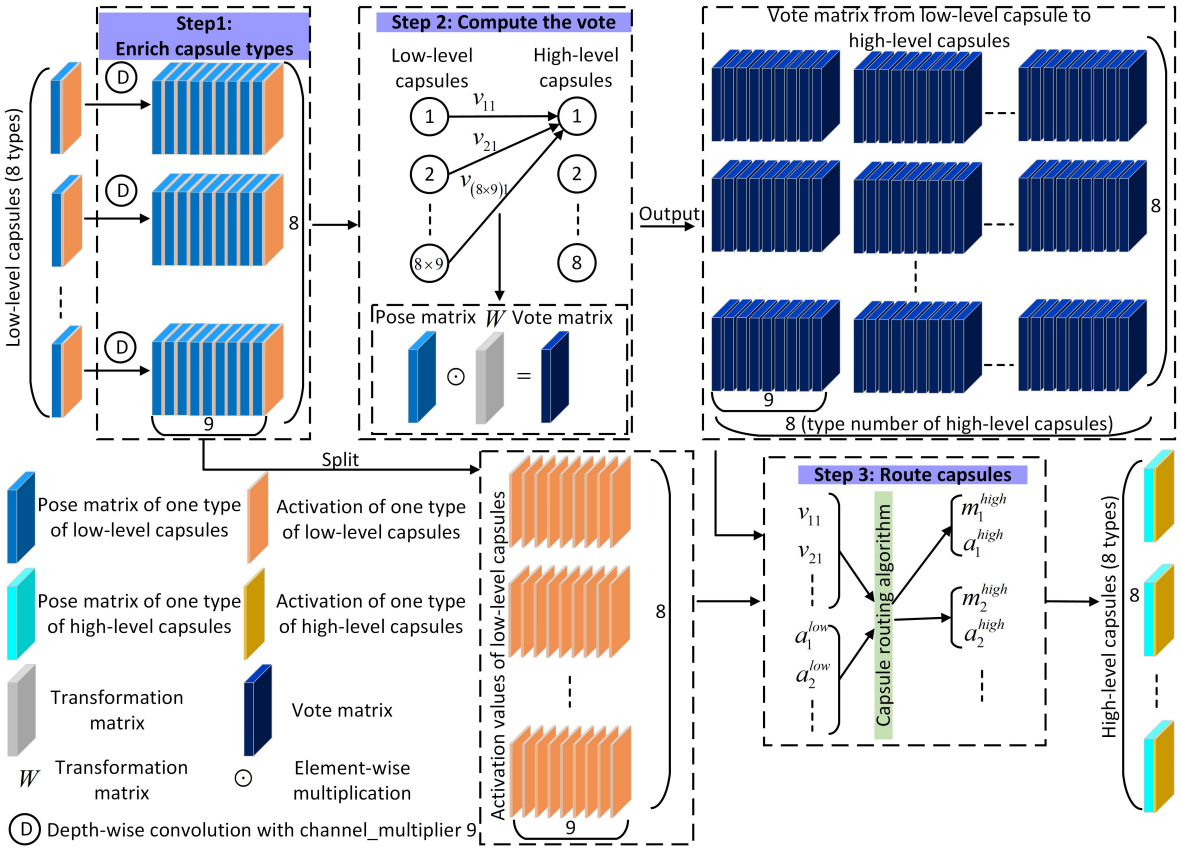


Fig. 9. Process of PORTNet by taking *ConvCaps1* as an example. In the figure, v_{ij} represents the vote matrix from capsule i in one type of low-level capsules to capsule j in one type of high-level capsules. m_*^{low}/a_*^{low} and m_*^{high}/a_*^{high} are pose matrices/activation values of low-level and high-level capsules, respectively.

each being elaborated by taking *ConvCaps1* for example in the following. Fig. 9 provides the process of PORTNet under *ConvCaps1*.

Step 1: Enrich capsule types. The 8-type capsules of each group ($88 \times 88 \times 8 \times 17$) are reshaped into $88 \times 88 \times 136$. A depth-wise convolution with the stride of 2 and the channel_multiplier of 9 is performed on the output capsules of *PrimaryCaps* to learn more rich-type capsules, *i.e.*, $44 \times 44 \times 9 \times 136$, which are reshaped into $1936 \times 72 \times 17$. Not unnaturally, the vectorized pose matrices and activation values are $1936 \times 72 \times [1 : 16]$ and $1936 \times 72 \times [17]$, respectively, where $[\cdot]$ represents the number of channels along the corresponding dimension.

Step 2: Compute the votes of low-level capsules for the adjacently high-level capsules. The pose matrix $M_i \in R^{4\times4}$ of the capsule i in one layer is constructed by reshaping the vectorized pose matrix. Between capsule i in one layer and capsule j in the layer above, it learns a trainable transformation matrix $W_{ij} \in R^{4\times4}$ discriminatively. The vote V_{ij} of capsule i for capsule j is calculated by multiplying M_i and W_{ij}, *i.e.*,

$$V_{ij} = M_i W_{ij}. \tag{1}$$

By Eq. (1), the resulting votes are $1936 \times 72 \times 8 \times 16$, where 1936, 72, 8, and 16 are the number of capsules of one type, type number of low-level capsules, type number of high-level capsules, and dimension of vote matrix, respectively.

Step 3: Route capsules between adjacent layers. Capsule routing intends to assign low-level capsules (parts) to high-level capsules (wholes), which can be solved by finding tight clusters of the votes from parts for the wholes in the layer above. To achieve this, a capsule routing algorithm is used to update the probability, with which a part is assigned to a whole. Such an assignment is determined based on the proximity of the current part's vote to the votes of the other parts for the same whole. This routing algorithm derives segmentation based on the knowledge of familiar shapes, rather than just using low-level cues such as proximity or agreement on color.

Specifically, the votes and the activation values of low-level capsules are input into the iterative routing algorithm, which will calculate the pose matrices (*i.e.*, $44 \times 44 \times 8 \times 16$) and activation values (*i.e.*, $44 \times 44 \times 8 \times 1$), respectively. Next, they are concatenated to form the high-level capsules ($44 \times 44 \times 8 \times 17$). These obtained capsules are fed into *ConvCaps2* within the same stream. Details of TSPORTNet can be found in Table 1.

Deep insights into the two-stream strategy for the task of salient object detection. The original CapsNet may have noisy capsules assignments because of its fully-connected routing. Such inaccurate capsule-assignments may cause no damage for some visual tasks, such as image classification, but it becomes problematic for salient object detection as it would make background noises appear on the saliency map.

TABLE 1
Details of TSPORTNet. s: stride; c_m: channel multiplier.

Layer	Process	Input	Operation	Output
Details of capsules construction				
PrimaryCaps	S1: Downsample feature maps	Feature maps of FLNet ($352 \times 352 \times 128$)	2Conv (s=2), ReLU	Downsampled feature maps ($88 \times 88 \times 16$)
	S2: Construct pose matrix	Downsampled feature maps ($88 \times 88 \times 16$)	Conv (s=1), reshape	Pose matrices ($88 \times 88 \times 16 \times 16$)
	S3: Construct activation	Downsampled feature maps ($88 \times 88 \times 16$)	Conv (s=1), reshape	Activation ($88 \times 88 \times 16 \times 1$)
	S4: Construct capsules	Pose matrices ($88 \times 88 \times 16 \times 16$) Activation ($88 \times 88 \times 16 \times 1$)	Concatenate	16-type capsules ($88 \times 88 \times 16 \times 17$)
	S5: Construct two groups	16-type capsules ($88 \times 88 \times 16 \times 17$)	Split	8-type capsules ($88 \times 88 \times 8 \times 17$) 8-type capsules ($88 \times 88 \times 8 \times 17$)
Details of PORTNet within each stream				
ConvCaps1	S1: Enrich the features of capsules	One group of primary capsules ($88 \times 88 \times 8 \times 17$)	Reshape, Depth conv (c_m=9, s=2), Reshape	$1936 \times 72 \times 17$
	S2: Compute votes of low-level capsules for the adjacently high-level capsules	Reshaped pose matrix ($1936 \times 72 \times [1:16]$) Transformation matrx $W(1936 \times 72 \times 16)$	Multiply	Votes ($1936 \times 72 \times 8 \times 16$)
	S3: Assign parts (low-level capsules) to wholes (high-level capsules)	Votes($1936 \times 72 \times 8 \times 16$) Reshaped activation ($1936 \times 72 \times 1$)	EM routing algorithm	Pose matrix ($44 \times 44 \times 8 \times 16$) Activation ($44 \times 44 \times 8 \times 1$)
ConvCaps2	S1: Enrich the features of capsules	Output capsules of *ConvCaps1* ($44 \times 44 \times 8 \times 17$)	Reshape, Depth conv (c_m=9, s=1), Reshape	$1936 \times 72 \times 17$
	S2: Compute votes of low-level capsules for the adjacently high-level capsules	Reshaped pose matrix ($1936 \times 72 \times [1:16]$) Transformation matrix $W(1936 \times 72 \times 16)$	Multiply	Votes ($1936 \times 72 \times 4 \times 16$)
	S3: Assign parts (low-level capsules) to wholes (high-level capsules)	Votes ($1936 \times 72 \times 4 \times 16$) Reshaped activation ($1936 \times 72 \times 1$)	EM routing algorithm	Pose matrix ($44 \times 44 \times 4 \times 16$) Activation ($44 \times 44 \times 4 \times 1$)
Details of capsule classification				
ClassCaps	S1: Integrate two streams	Pose matrices of two streams (both $44 \times 44 \times 4 \times 16$) Activation values of two streams (both $44 \times 44 \times 4 \times 1$)	Concatenate, reshape	Pose matrix ($1936 \times 8 \times 16$) Activation ($1936 \times 8 \times 1$)
	S2: Compute votes of low-level capsules for the adjacently high-level capsules	Reshaped pose matrix ($1936 \times 8 \times 16$) Transformation matrix $W(1936 \times 72 \times 16)$	Multiply	Votes ($1936 \times 8 \times 2 \times 16$)
	S3: Assign parts (low-level capsules) to wholes (high-level capsules)	$1936 \times 8 \times 2 \times 16$ (votes) $1936 \times 8 \times 1$ (reshaped activation)	EM routing algorithm	$44 \times 44 \times 2 \times 16$ (pose) $44 \times 44 \times 2 \times 1$ (activation) $44 \times 44 \times 2 \times 17$ (2-type capsules)

To address this issue, we propose a two-stream strategy to implement CapsNet via a locally-connected routing to detect the salient object. Such practice narrows the searching space and thus further lowers the likelihood of having noisy assignments. On top of that, it ensures that relevant salient parts can be routed together to form a complete salient object while suppressing background noise caused by misrouting background regions to salient parts in the saliency map. Some visual examples can be found in Fig. 14. In this context, the two-stream strategy fits better to the salient object detection than the original CapsNet as it gains both detection accuracy and detection efficiency.

3.2.3 Capsule Classification

Those more complete capsules obtained by the two streams are finally classified to be salient or background, which is implemented by a Class Capsule (*ClassCaps*) layer. The architecture of *ClassCaps* is similar to **Step 2** and **Step 3** in *ConvCaps1*. Through the *ClassCaps* layer, the capsules of two streams will be assigned to two types of capsules corresponding to the salient object and background, in which way some relevant parts will be clustered together to form a salient object. Considering the capsule activation values represent the existence probability of the entity, we take them as the output of *ClassCaps*, *i.e.*, $44 \times 44 \times 2$, to form the saliency prediction map of the input image. In other words, this is also the output of TSPORTNet.

3.3 Correlation-Aware Capsule Routing

Different from the image-level classification, salient object detection, a task of pixel-level dense segmentation, involves more complicated part-object relationships, given a large-scale image. This is challenging for the original capsule routing algorithm in [25] designed for small-size image classification. To remedy this issue, we resort to the capsule correlations for more powerful capsule routing.

As observed in Fig. 4, different types of capsules representing different patterns may change as the input image varies. However, pattern assignments calculated by the routing algorithm do not necessarily change and they somewhat relate to the correlations between different types of capsules across adjacent layers. Specifically, highly correlated capsules across adjacent layers share high familiarity, which will be more likely to be routed together. In light of this, we could take the correlations between different types of capsules of the preceding training samples as a capsule routing awareness to facilitate the subsequent training image.

Suppose x_I and x_J are the features of type-I capsules and type-J capsules from two consecutive layers, the correlation between them is encoded as:

$$c_{IJ} = 1 - Sigmoid(\|x_I - x_J\|_2^2). \tag{2}$$

We embed this capsule correlation information into the EM routing algorithm in [25] via a residual formulation

$$R_{ij}^{reg} = R_{ij} + R_{ij} \cdot c_{IJ}, i \in I, j \in J, \tag{3}$$

where R_{ij} is the capsule assignment probability in [25]. i and j are two capsules belonging to type-I and type-J, respectively.

The resultant correlation-aware capsule routing algorithm is scheduled in Algorithm 1, where the two-stream strategy is also exhibited. Algorithm 1 consists of five steps, *i.e.*, initialization, M-procedure, E-procedure, capsule correlations embedding, and capsule correlations calculation. We will explain these steps in the following.

Algorithm 1 Correlation-aware capsule routing algorithm. a and V are the activation values and vote matrices, respectively. Ω^{high} is the type number of high-level capsules. I and J represent two types of capsules across adjacent layers. i and j are two capsules belonging to type-I and type-J, respectively. β_u and β_a are trained discriminatively. λ is an inverse temperature parameter increased by 1 after each routing iteration. h represents one component of one vector.

Procedure Correlation-aware capsule routing (a, V)

1. Initialization:

$R_{ij} = \frac{1}{\Omega^{high}}$

$c_{IJ} = 0$

for n images **do**

  for m streams **do**

    for t iterations **do**

      2. M-procedure for high-level capsule j:

$\forall i \in \Omega_L : R_{ij} = R_{ij} \cdot a_i$

$\forall h : \mu_j^h = \frac{\sum_i R_{ij} V_{ij}^h}{\sum_i R_{ij}}$

$\forall h : \left(\sigma_j^h\right)^2 = \frac{\sum_i R_{ij} \left(V_{ij}^h - \mu_j^h\right)^2}{\sum_i R_{ij}}$

$cost^h = \left(\beta_u + \log\left(\sigma_j^h\right)\right) \sum_i R_{ij}$

$a_j = logistic\left(\lambda\left(\beta_a - \sum_h cost^h\right)\right)$

      3. E-procedure for low-level capsule i:

$\forall j \in \Omega_{L+1} : p_j = \frac{\exp\left(-\sum_h^H \frac{\left(V_{ij}^h - \mu_j^h\right)^2}{2\left(\sigma_j^h\right)^2}\right)}{\sqrt{\prod_h^H 2\pi\left(\sigma_j^h\right)^2}}$

$\forall j \in \Omega_{L+1} : R_{ij} = \frac{a_j p_j}{\sum_{k \in \Omega_{L+1}} a_k p_k}$

      4. Capsule correlations embedding (removed at the test stage):

$R_{ij} = R_{ij} + R_{ij} \cdot c_{IJ}, i \in I, j \in J$

    end

  end

  5. Capsule correlations calculation:

$c_{IJ} = Sigmoid(1 - \|x_I - x_J\|_2^2)$

end

Step 1: Initialization. We initialize the assignment probability R_{ij} to be uniformly distributed, *i.e.*, low-level capsules equally related to any high-level capsule. Besides, c_{IJ} is initialized to be zero.

Step 2: M-procedure for high-level capsule j. M-procedure computes an updated Gaussian model (μ, σ) and the activation a_j of high-level capsule j from the activation a_i, the current R_{ij}, and votes V.

Step 3: E-procedure for low-level capsule i. E-procedure determines the assignment probability R_{ij} of each low-level capsule to a high-level capsule based on the new (μ, σ) and the activation a_j.

Step 4: Capsule correlations embedding. We embed the capsule correlations into the assignment probability with the aim to assign highly correlated capsules together.

Step 5: Capsule correlations calculation. After training each image, we calculate the capsule correlations between different types of capsules across adjacent layers.

R_{ij}^{reg} contains two types of information: 1) assignment probability information between capsule i and capsule j for the current training image; 2) correlation information between type-I capsules and type-J capsules of all the preceding training images. Due to the involvement of capsule correlations, R_{ij}^{reg} is able to activate those high-familiarity capsules while alleviating confusing capsule assignments, which is conductive to exploring accurate part-object relationships. Also, the training procedure will be sped up with the guidance of capsule correlations.

Deep insights into the correlation-aware capsule routing algorithm to salient object detection. The capsule routing algorithm is designed to solve the part-object routing problem, *i.e.*, finding relevant instead of individual parts of the same salient object based on the familiar relationships between the salient object and its parts. It eventually helps to segment out the whole salient object by forming all the relevant salient parts to compose a complete salient object. This is superior to the previous salient object detectors that segment out the salient object by finding individual salient regions, which easily leads to the incomplete segmentation. To enhance this capability of exploring the familiar part-object relationships, we propose a correlation-aware routing algorithm by taking into consideration the correlation between lower-layer capsules (parts) and higher-layer capsules (wholes). Doing so enhances the capsule assignments between two similar types of capsules while suppressing those between two irrelevant ones, which further helps to improve the familiar relationships between the salient parts and their whole salient object. Consequently, the saliency map will wholly identify the salient object, as can be verified in Fig. 15.

3.4 WGNet

By exploring part-object relationships, TSPORTNet can infer the whole salient object by finding salient parts, generating a Capsule Wholeness Map (CWM). We take the salient capsule output by TSPORTNet as a CWM. Fig. 5 provides some CWM examples. However, there still exist some problems with the CWM: i) Blurriness occurs around the object boundary; ii) Some bright spots emerge in the background region; iii) Poor smoothness appears on the salient object surface. To tackle these problems, we design a Wholeness Guidance Network (WGNet), where the CWM is used to guide multi-level feature maps of FLNet. On the one hand, multi-level features will learn more accurate saliency cues with the guidance of the CWM. On the other hand, the CWM will get a smoother surface with the aid of rich saliency cues of those feature maps of FLNet. The detailed architecture of WGNet can be found in Fig. 6. To be specific, for each level of feature maps $\mathrm{FM}i (i = 1, 2, 3, 4)$, the feature maps of WGNet can be formulated as

$$F_i^{WGNet} = Conv\left(Concat\left(Up\left(O_{CWM}\right), \mathrm{FM}i\right)\right), \quad (4)$$

where $Conv(\cdot)$, $Concate(\cdot)$, and $Up(\cdot)$ represent the operations of convolution, concatenation, and upsampling, respectively. F_i^{WGNet} is further used to achieve the saliency prediction S_i for the stage of i by

$$S_i = Conv\left(Up\left(F_i^{WGNet}\right)\right). \quad (5)$$

It is noted that S_1 is achieved by directly performing convolution on FM1 owing to a large resolution of FM1.

Furthermore, the capsule wholeness map F_{CWM} is used to compute the saliency map S_5 by

$$S_5 = Conv\left(Up\left(F_{CWM}\right)\right). \tag{6}$$

The final saliency map is achieved by combining $\{S_i\}\,(i = 1, 2, 3, 4, 5)$ by

$$S = S_1 + S_2 + S_3 + S_4 + S_5. \tag{7}$$

Unlike the existing spatial attention maps [60–62] that fixate at the location of the salient object, the proposed capsule wholeness map can not only locate the salient object but also provide the object wholeness.

Additionally, our framework is distinct from the coarse-to-fine methods [63, 64] that first generate a coarse saliency map and then apply one or more refinement stages to restore those missing details, *e.g.*, object boundaries, salient context, *etc*. The main difference lies in that our wholeness map predicted by TSPORTNet can detect the whole salient object with fewer object parts missed.

3.5 Saliency Inference

We adopt a joint loss function by combining the cross-entropy loss function used in [43] and the IoU boundary loss function to train the proposed salient object detection network.

The cross-entropy loss function is formulated as

$$CE\left(\mathbf{v}\right) = -\frac{1}{N}\sum_{p=1}^{N}\sum_{c\in\{0,1\}}\left(y(\mathbf{v}_p) = c\right)\left(\log\left(\hat{y}(\mathbf{v}_p) = c\right)\right), \tag{8}$$

where $\mathbf{v}_p$ represents the location of pixel p. $y(\mathbf{v}_p)$ and $\hat{y}(\mathbf{v}_p)$ represent saliency values of the pixel p in the ground truth and the predicted saliency map, respectively.

The IoU boundary loss function is defined as

$$IoU\ \ Loss = 1 - \frac{2|C_l \cap \hat{C}_l|}{|C_l| + |\hat{C}_l|}, \tag{9}$$

where $\hat{C}_l$ and C_l are the gradient magnitudes of saliency map and ground truth corresponding to region l. The gradient magnitude is computed by using a Sobel operator followed by a tanh activation on the saliency map. $|\cdot|$ represents the number of non-zero entities in a mask.

The joint loss function combines the cross-entropy loss function and the IoU Boundary loss function, *i.e.*,

$$Joint\ \ Loss = CE + IoU\ \ Loss. \tag{10}$$

4 Insight Analysis

4.1 Salient Property of Part-Object Relationships

The property of part-object relationships for salient object detection is derived from the idea that two parts will be clustered together to form a whole object if they share familiar properties. In other words, two capsules i and k will be put together to make the capsule j in the layer above, if

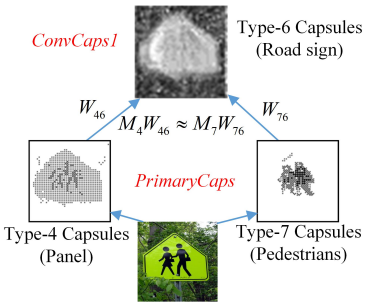


Fig. 10. Illustrations for the part-object relationships. Type-4 capsules (panel) and type-7 capsules (pedestrians) make up type-6 capsules (a whole object) in the higher *ConvCaps1* layer based on their approximately equal votes to the higher capsule. Here, we visualize the capsule activation values since they are the saliency probabilities of the entities.

$$M_i W_{ij} \approx M_k W_{kj}. \tag{11}$$

To give a clear insight for the property of part-object relationships employed in salient object detection, we visualize the intermediate layers of a real example (as shown in Fig. 10) based on a Single-Stream PORTNet (SSPORTNet), which is a baseline network by directly adopting the original CapsNet after FLNet. Two observations can be viewed from Fig. 10: 1) Type-4 capsules and type-7 capsules in the *PrimaryCaps* layer do capture two parts, *i.e.*, pedestrians and panel, while type-6 capsules in the higher *ConvCaps1* layer clearly depict the whole object; 2) Due to approximated votes ($M_4W_{46} \approx M_7W_{76}$), type-4 capsules and type-7 capsules capturing different parts make up the higher type-6 capsules representing a complete object, *i.e.*, road sign. Hence, the natural capability of PORTNet in modeling part-object relationships can detect the whole salient object by finding familiar object parts, thereby addressing the object part missing problem that the CNNs based saliency detectors are suffering from.

4.2 In Comparison to CapsNet

Our proposed framework differs the original CapsNet in four aspects.

Computational complexity: The original CapsNet votes each lower capsule to all types of higher capsules, yielding a heavy computational complexity. Differently, our two-stream strategy assigns each lower capsule to one stream types of higher capsules only, instead of all types of capsules at the higher layer. In doing so, our strategy learns transformation matrices 4 times fewer than those in the original CapsNet, thus decreasing the computational complexity of the capsule routing algorithm dramatically. Such a significant reduction of computation helps CapsNet to deal with complicated dense prediction applications such as salient object detection.

Feature learning: Unlike the original CapsNet that uses (Conv + ReLU) layers for feature extraction, we utilize a FLNet to learn more primitive features for TSPORTNet. This will improve the performance by a large margin, which is verified in the experiment part.

Capsule classification: The original CapsNet classifies capsules to several capsules corresponding to image categories, which is not suitable for the dense prediction task.

Instead, since salient object detection is essentially a binary segmentation task, we classify the capsule at each pixel position to two capsules, including salient capsule and background capsule.

Capsule routing algorithm: To enhance the original capsule routing algorithm, we take the correlations between different capsule types across adjacent layers from the preceding training samples into consideration. Doing so helps to activate those high-familiarity capsules while alleviating confusing capsule assignments, thereby accelerating the training procedure and gaining more accurate part-object relationships.

4.3 Analysis of CWM

To better exhibit the superiority of CWM in WGNet, we replace CWM with a Convolutional Map (ConvM) that is obtained by replacing *ConvCaps 1* and *ConvCaps 2* with two convolutional layers. Toy examples are shown in Fig. 11. In Fig. 11, (b) and (d) visualize ConvM and CWM with the resolution of 44×44; (c) and (e) are the side outputs (*i.e.*, S5 in Fig. 6) with the resolution of 352×352, which are obtained by directly conducting several deconvolutional layers on ConvM and CWM, respectively. Comparing Fig. 11(b) and (d), we can observe that ConvM in Fig. 11(b) hardly identifies the salient object with too much background noises, while CWM in Fig. 11(d) can segment out the whole salient object discriminatively. As a result, the side output of CWM in Fig. 11(e) can detect the whole salient object with clean background. This is beneficial from the part-object relationships explored by TSPORTNet. More analyses will be presented in the following Sec. 5.4.6.

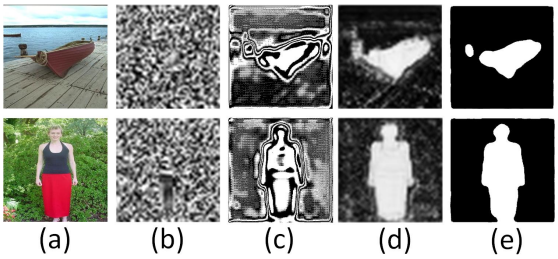


Fig. 11. Illustrations for the superiority of CWM. (a) Images; (b) ConvM; (c) Saliency map inferred by ConvM; (d) CWM; (e) Saliency map inferred by CWM. (b) and (d): 44×44; (c) and (e): 352×352.

5 Experiment and Analysis

In this section, numerous experiments and analyses are conducted to verify the effectiveness and superiorities of our proposed deep salient object detection network.

5.1 Benchmark Datasets

We evaluate the performance of our model on five benchmark datasets, details of which are described as follows.

ECSSD [65] contains 1000 images with complicated structures, which are collected from the Internet. **HKU-IS** [17] consists of 3000 training images and 1447 test images, which are with multiple disconnected objects. **PASCAL-S** [66] includes 850 images in various scenes, which are collected from the validation dataset of PASCAL VOC 2010 [67] segmentation challenge. **DUTS** [68] contains 10533 training images and 5019 test images, which are with different scenes and various sizes. **DUT-OMRON** [13] has 5168 images with different sizes and complex structures. In terms of HKU-IS [17] and DUTS [68], only the test images are used for evaluations in our experiments.

5.2 Evaluation Criteria

We evaluate the performance of our model as well as other state-of-the-art methods from both visual and quantitative perspectives. The quantitative metrics include Precision Recall (PR), F-measure, Mean Absolute Error (MAE), S-measure, and E-measure. Given a continuous saliency map, a binary mask B is achieved by thresholding. Precision is defined as $Precision = |B \cap G|/|B|$, and recall is defined as $Recall = |B \cap G|/|G|$, where G is the corresponding ground truth. A PR curve is plotted under different thresholds.

F-measure is an overall performance indicator, which is computed by

$$F_\beta = \frac{\left(1+\beta^2\right) Precision \times Recall}{\beta^2 Precision + Recall}. \tag{12}$$

As suggested in [69], $\beta^2 = 0.3$.

MAE is defined as

$$MAE = \frac{1}{W \times H} \sum_{i=1}^{W} \sum_{j=1}^{H} |S\left(i,j\right) - G\left(i,j\right)|, \tag{13}$$

where W and H are the width and height of the image, respectively.

S-measure (S_m) [70] is computed by

$$S_m = \alpha S_o + \left(1-\alpha\right) S^r, \tag{14}$$

where S_o and S_r represent the object-aware and region-aware structure similarities between the prediction and the ground truth, respectively. α is set to 0.5 [70].

E-measure (E_m) [71] combines local pixel values with the image-level mean value to jointly evaluate the similarity between the prediction and the ground truth.

5.3 Implementation Details

The proposed model is implemented in Tensorflow [72]. To avoid over-fitting caused by training from scratch, the five stacked convolutional layers in FLNet are initialized by Conv1_2, Conv2_2, Conv3_3, Conv4_3, and Conv5_3 of the pretrained VGG16 model [73], respectively. The other weights are initialized randomly with a truncated normal ($\sigma = 0.01$), and the biases are initialized to 0. The Adam optimizer [74] is used to train our model with an initial learning rate of 10^{-5}, $\beta_1 = 0.9$, and $\beta_2 = 0.999$. The training dataset of DUTS [75] is chosen as the training dataset with horizontal flipping as the data augmentation technique.

TABLE 2
Comparisons of F_β and MAE values for different ablation analyses on ECSSD [65]. Especially in (d), "w/ CAR" and "w/o CAR" represent the proposed correlation-aware capsule routing algorithm and the original capsule routing algorithm adopted in [25], respectively.

Section	Framework versions	Metric	
		F_β	MAE
Training with the cross-entropy loss function.			
(a) Sec. 5.4.1	FLNet + TSPORTNet	**0.8873**	**0.0515**
	Conv + ReLU + TSPORTNet	0.6545	0.1504
(b) Sec. 5.4.2	FLNet + TSPORTNet	**0.8873**	**0.0515**
	FLNet	0.8250	0.0694
(c) Sec. 5.4.3	FLNet + TSPORTNet	**0.8873**	**0.0515**
	FLNet + SSPORTNet	0.8706	0.0644
Training with the joint loss function.			
(d) Sec. 5.4.4	FLNet+TSPORTNet (w/ CAR)	**0.9098**	**0.0437**
	FLNet+TSPORTNet (w/o CAR)	0.9007	0.0503
(e) Sec. 5.4.5	+WGNet	**0.9135**	**0.0417**
	-WGNet	0.9098	0.0437
(f) Sec. 5.4.6	CWM	**0.9135**	**0.0417**
	ConvM	0.8680	0.0572
Training "FLNet + TSPORTNet" with different loss functions.			
(g) Sec. 5.4.7	$CE + IoU$	**0.9007**	**0.0503**
	CE	0.8873	0.0515

5.4 Ablation Analysis

5.4.1 FLNet

To explore the validity of FLNet, we compare two framework versions, including "FLNet + TSPORTNet" and "Conv + ReLU + TSPORTNet" that learn features of the input image through (Conv + ReLU) layers used by the original CapsNet. The performance comparisons are given in Table 2(a) and Fig. 12. Obviously, in Table 2(a), FLNet promotes the performance significantly, specifically 23.28 and 9.89 points in terms of F_β and MAE values, respectively. This evidently proves that rich deep features of the input image fed into CapsNet play an important role in the final dense prediction. More visibly in Fig. 12, thanks to the rich features learned by FLNet, the network can identify low-contrast salient parts, generating a whole salient object. This firmly confirms that rich features of the input image can help CapsNet find those tight low-contrast parts to compose a whole object.

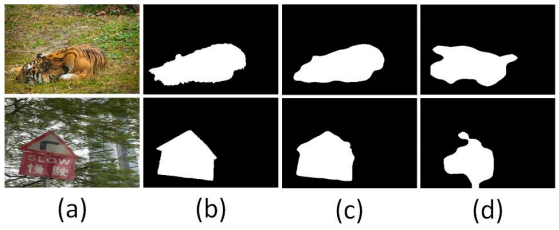


Fig. 12. Visual comparisons for FLNet. (a) Image; (b) GT; (c) FLNet + TSPORTNet; (d) Conv + ReLU + TSPORTNet.

5.4.2 TSPORTNet

To demonstrate the effectiveness of TSPORTNet, we compare two baselines, *i.e.*, "FLNet + TSPORTNet" and "FLNet". Table 2(b) and Fig. 13 describe the detailed performance comparisons from the quantitative and qualitative perspectives, respectively. It can be found in Table 2(b) that TSPORTNet improves the performance by a clear margin, specifically 6.23 and 1.79 points for F_β and MAE values, respectively. Similar conclusions can be observed in Fig. 13, where FLNet tends to miss those locally low-contrast salient parts, *e.g.*, the yellow region within the local diamond panel in the top row of Fig. 13, or those globally low-contrast salient parts, *e.g.*, the birds in the entire image in the bottom row of Fig. 13. By contrast, through exploring the part-object relationships, TSPORTNet grabs the whole salient object with all the salient parts detected.

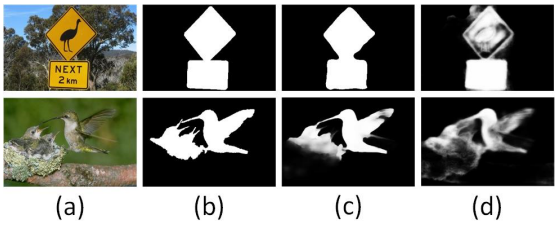


Fig. 13. Visual comparisons for TSPORTNet. (a) Image; (b) GT; (c) FLNet + TSPORTNet; (d) FLNet.

5.4.3 Two-Stream Strategy

To better understand the superiority of the two-stream strategy, we investigate two architectures, including "FLNet + TSPORTNet" and "FLNet + SSPORTNet", where the latter is implemented by directly adopting the original CapsNet following FLNet. Table 2(c) and Fig. 14 show the quantitative and visual comparisons, respectively. As observed in Table 2(c), compared with SSPORTNet, TSPORTNet greatly elevates the performance by 1.67 and 1.29 points with regard to F_β and MAE values, respectively. Also, as illustrated in Fig. 14, SSPORTNet mislabels some background regions as parts of the salient object, which indicates SSPORTNet introduces some noisy capsule assignments. In contrast to that, due to the involvement of the two-stream strategy, our TSPORTNet successfully alleviates those noisy capsule assignments, helping to cluster correct salient parts together to compose the whole salient object.

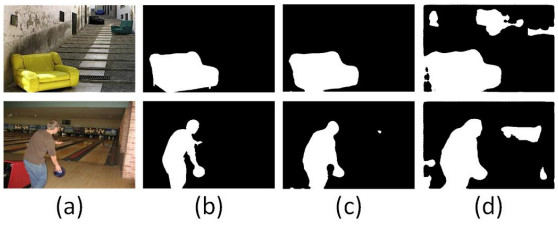


Fig. 14. Visual comparisons for the two-stream strategy. (a) Image; (b) GT; (c) FLNet + TSPORTNet; (d) FLNet + SSPORTNet.

5.4.4 Correlation-Aware Capsule Routing Algorithm

To show the effectiveness of the correlation-aware capsule routing algorithm, we explore the difference between "FLNet + TSPORTNet (w/ CAR)" and "FLNet + TSPORTNet (w/o CAR)", where "CAR" represents the correlation-aware capsule routing algorithm. To this end, we compare two frameworks using the proposed correlation-aware capsule routing algorithm (*i.e.*, "FLNet + TSPORTNet (w/ CAR)") and the original capsule routing algorithm (*i.e.*,

"FLNet + TSPORTNet (w/o CAR)") adopted in [25], respectively. The performance comparisons can be found in Table 2(d), which shows that the correlation-aware capsule routing algorithm enhances the performance by 0.91 and 0.66 points in terms of F_β and MAE, respectively. Fig. 15 exhibits two complex scenes to illustrate the superiority of the correlation-aware capsule routing algorithm. Obviously, the correlation-aware capsule routing algorithm helps to grab more satisfactory object wholeness. This is beneficial from the involvement of capsule correlations, which activate high-familiarity capsules between adjacent layers, thereby exploring more accurate part-object relationships.

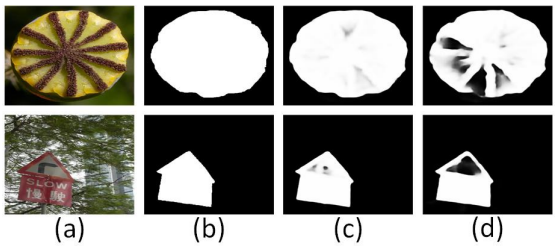


Fig. 15. Visual comparisons for the correlation-aware capsule routing algorithm. (a) Image; (b) GT; (c) FLNet + TSPORTNet (w/ CAR); (d) FLNet + TSPORTNet (w/o CAR). "w/ CAR" and "w/o CAR" represent the proposed correlation-aware capsule routing algorithm and the original capsule routing algorithm adopted in [25], respectively.

As illustrated in Fig. 16, the correlation-aware routing algorithm leads to a faster convergence than the original capsule routing algorithm, which makes the network training a lot easier. Concretely, when the loss converges to 0.05, the correlation-aware capsule routing algorithm will save 4 epochs covering 24 hours, which is a significant acceleration of the training procedure.

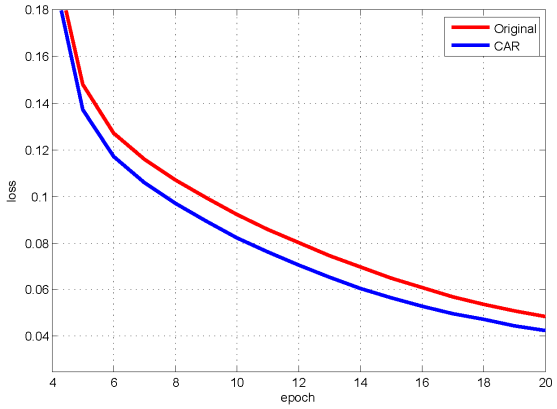


Fig. 16. Training loss with the cross-entropy loss function. "Original" and "CAR" mean the losses obtained by the original capsule routing algorithm in [25] and the correlation-aware capsule routing algorithm, respectively. When the loss converges to 0.05, our CAR saves 4 epochs covering 24 hours.

5.4.5 WGNet

To illustrate the effectiveness of WGNet, we compare the entire framework with a modified version that is obtained by replacing WGNet with three deconvolution-ReLU operations and one convolution on the capsule wholeness map to infer the saliency map. Table 2(e) and Fig. 17 give detailed

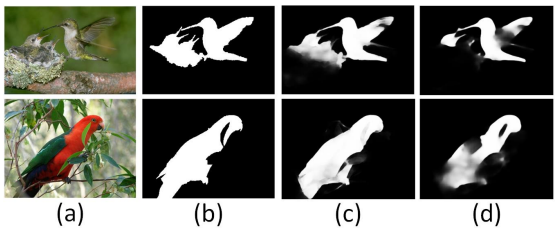


Fig. 17. Visual comparisons for WGNet. (a) Image; (b) GT; (c) +WGNet; (d) -WGNet.

comparisons for these two versions. It is clear in Table 2(e) that WGNet increases F_β and decreases MAE by 0.37 and 0.20 points, respectively. In Fig. 17, we can find that WGNet helps to get more clear object boundaries as well as more uniform foreground and background maps. This proves that the aggregation of the capsule wholeness map and multi-level features facilitates to generate the final saliency map.

5.4.6 CWM

To investigate the superiority of the CWM, we replace CWM with a Convolutional Map (ConvM) that is obtained by replacing *ConvCaps 1* and *ConvCaps* 2[4] with two convolutional layers. Table 2(f) and Fig. 18 exhibit the performance comparisons between our CWM and ConvM in quantitative and visual manners, respectively. It is obvious from Table 2(f) that our CWM achieves a significant performance gain compared to ConvM, *e.g.*, 4.55 and 1.55 points in terms of F_β and MAE values, respectively. Additionally, in Fig. 18, different from ConvM that misses some salient object parts, our CWM can effectively produce the whole salient object. These comparisons demonstrate that our CWM that grabs the object wholeness is superior to ConvM.

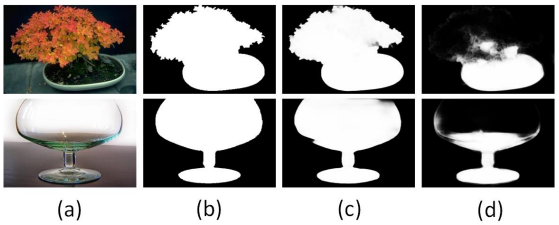


Fig. 18. Visual comparisons for the capsule wholeness map. (a) Image; (b) GT; (c) CWM; (d) ConvM.

5.4.7 Loss Function

To compare the performance of different training loss functions, we train the baseline model, *i.e.*, "FLNet + TSPORTNet", using the cross-entropy loss function and the joint loss function, respectively. Table 2(g) and Fig. 19 show the quantitative and qualitative comparisons. It is obvious from Table 2(g) that the joint loss function can promote the performance to some extent, specifically 1.34 and 0.12 points for F_β and MAE, respectively. As shown in Fig. 19, the joint loss function helps to achieve clear object boundaries.

4. Here, *ConvCaps 1* and *ConvCaps 2* correspond to the two convolutional capsule layers, which are illustrated in Fig. 6.

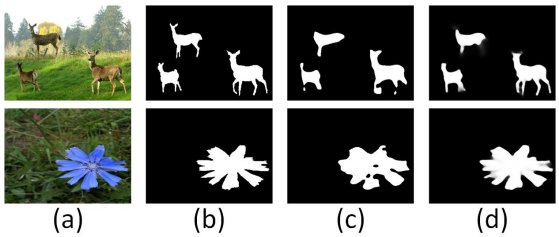


Fig. 19. Visual comparisons for different loss functions. (a) Image; (b) GT; (c) Saliency map obtained by using the cross-entropy loss function; (d) Saliency map obtained by using the joint loss function.

5.4.8 Visualization of Different Phases

To illustrate the roles of different phases in our method, we visualize the detection results of different phases on the same image, which are shown in Fig. 20. As can be seen, FLNet indeed captures rich features of the image, and TSPORTNet filters out background noises, thus promoting the salient regions via exploring the part-object relationships. The correlation-aware routing algorithm helps to capture the whole salient object by computing more accurate capsule assignments. With the aid of WGNet, the entire model can detect the whole salient object with uniform saliency values.

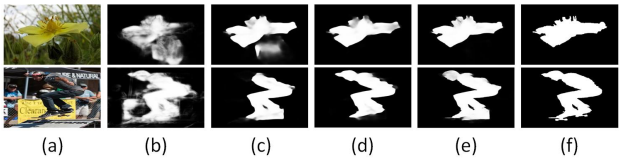


Fig. 20. Detection results of different phases. (a) Image; (b) FLNet; (c) FLNet + TSPORTNet; (d) FLNet + TSPORTNet + CAR; (e) Our entire model; (f) GT.

5.5 Comparison with the State-of-the-Art Methods

In this section, we compare our method with 14 state-of-the-art methods, including TSPOANet [26], JointCRF [76], ToHR [44], PoolNet [45], CPD [78], AFNet [79], BMP [43], LFR [80], Amulet [41], UCF [81], DLS [21], ELE [22], ELD [82], and MDF [17]. Visual and quantitative comparisons are both taken into account for fair comparisons.

5.5.1 PR Curve and F-measure Curve

Fig. 21 plots PR and F-measure curves of our method against the state-of-the-art methods on 4 popular salient object detection datasets. If looking at the upper right corners of the PR curves, our method produces higher precision when the recall score is close to 1, which indicates a lower false positive rate. Besides, it can be found that our method beats most of the others in the two upper corners of the F-measure curves. This demonstrates that on our saliency maps, background regions achieve uniformly low values whilst salient regions are provided with uniformly high values. These observations from the PR and F-measure curves also indicate that the saliency maps of our framework are much closer to the ground truth.

5.5.2 Quantitative Comparisons

Table 3 lists the average F-measure, MAE, S-measure, and E-measure values of different methods. It can be easily seen from Table 3 that our model achieves competitive performance with the state-of-the-art methods on five benchmarks. Specifically, over four metrics on five benchmarks, we obtain 7 top-1, 15 top-2, and 19 top-3 places, respectively. The second best method, *i.e.*, PoolNet [45], gets 7 top-1, 10 top-2, and 17 top-3 places, respectively. The third best method, *i.e.*, CPD [78], obtains 6 top-1, 13 top-2, and 14 top-3 places, respectively. This clearly demonstrates the superiority of the part-object relational property adopted by our framework for salient object detection. We are especially excited to obtain a noticeable gain over the previous version TSPOANet [26], which indicates the performance improvements caused by the correlation-aware routing algorithm and the capsule wholeness map guidance network.

5.5.3 Visual Comparison

To further explain the superiority of our proposed approach, Fig. 22 displays saliency maps of some images with various challenging properties, including small objects, large objects, low contrast, low compactness, multiple objects, center bias, complex scenes. Each image usually incorporates multiple properties. For the first group of images, the compared methods mostly introduce some background noises. For the second group, most existing methods usually miss some salient object parts. For the third group, existing saliency detectors mostly miss some salient objects or get poor object smoothness. For the last group, the compared approaches mostly miss some salient parts. Instead, taking all circumstances into account, our model highlights the right salient object, and produces good wholeness for those images with various properties. To sum up, compared with the state-of-the-arts, our detection results are closer to the ground truth in various cases.

5.5.4 Time Analysis

Table 4 lists the running time of some methods (with one NVIDIA 1080Ti GPU). It can be observed that our method is much faster than CapsNet [25], JointCRF [76], ToHR [44], and MDF [17]. In contrast, our approach is slower than BMP [43], LFR [80], Amulet [41], and UCF [81]. This is mainly attributed to the computationally expensive EM routing algorithm adopted in our model. However, notice that our method achieves much better performance than these methods, as evidenced in Fig. 21 and Table 3. Specifically, as illustrated in Fig. 22, we get much better wholeness and uniformity for the saliency map than these state-of-the-art methods.

Notably, compared with the original CapsNet, our method can greatly accelerate the running speed by 65%, which benefits from the use of the proposed two-stream strategy. Specifically, our method takes 0.35 seconds to process one image with cropped size 352×352, which is feasible for some applications, *e.g.*, medical diagnosis, image/video editing and rotoscoping on mobile.

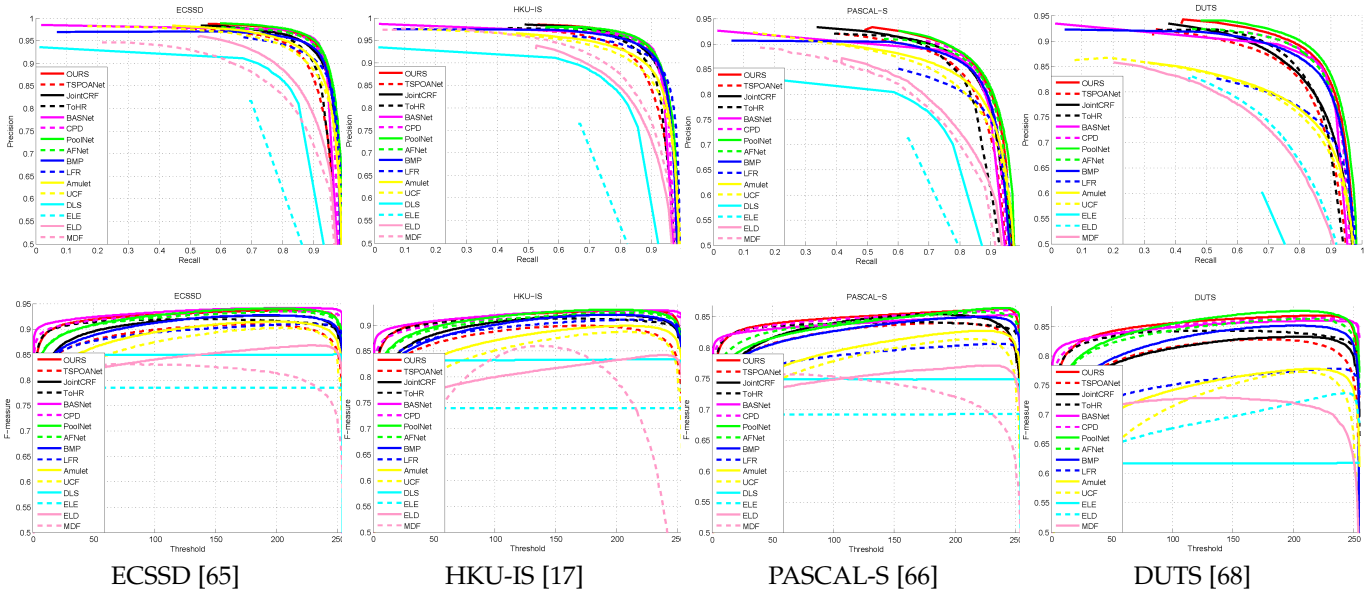


Fig. 21. PR and F-measure curves of different methods on four popular salient object detection datasets.

TABLE 3
F_β, MAE, S_m, and E_m values of different methods. Top three methods are marked by red, blue, and green, respectively. "-" means that the corresponding authors do not provide the detection results of the dataset.

	ECSSD [65]				HKU-IS [17]				PASCAL-S [66]				DUTS [68]				DUT-OMRON [13]			
	F_β	MAE	S_m	E_m	F_β	MAE	S_m	E_m	F_β	MAE	S_m	E_m	F_β	MAE	S_m	E_m	F_β	MAE	S_m	E_m
OURS	0.9135	0.0410	0.9129	0.9229	0.9010	0.0324	0.9091	0.9502	0.8204	0.0707	0.8498	0.8578	0.8092	0.0433	0.8707	0.8877	0.7436	0.0579	0.8230	0.8557
TSPOANet [26]	0.8873	0.0515	0.8684	0.9020	0.8795	0.0391	0.8656	0.9263	0.8123	0.0749	0.8142	0.8508	0.7971	0.0482	0.8202	0.8748	0.7030	0.0628	0.7692	0.8232
JointCRF [76]	0.8956	0.0493	0.9068	0.9152	0.8817	0.0394	0.9032	0.9384	0.7898	0.0824	0.8410	0.8368	0.7444	0.0588	0.8358	0.8477	0.7379	0.0574	0.8207	0.8571
ToHR [44]	0.9023	0.0544	0.8829	0.9171	0.8923	0.0420	0.8827	0.9357	0.8008	0.0855	0.8068	0.8465	0.7932	0.0512	0.8291	0.8835	0.7079	0.0660	0.7718	0.8411
BASNet [77]	0.9023	0.0544	0.8829	0.9171	0.8923	0.0420	0.8827	0.9357	0.8008	0.0855	0.8068	0.8465	0.7932	0.0512	0.8291	0.8835	0.7079	0.0660	0.7718	0.8411
CPD [78]	0.9145	0.0402	0.9102	0.9216	0.8962	0.0332	0.9045	0.9471	0.8199	0.0721	0.8446	0.8563	0.8132	0.0429	0.8666	0.8921	0.7452	0.0567	0.8177	0.8633
PoolNet [45]	0.9098	0.0417	0.9173	0.9205	0.8942	0.0333	0.9118	0.9477	0.8060	0.0716	0.8518	0.8452	0.7986	0.0419	0.8787	0.8807	0.7386	0.0562	0.8324	0.8574
AFNet [79]	0.9076	0.0418	0.9134	0.9180	0.8891	0.0355	0.9058	0.9424	0.8149	0.0717	0.8482	0.8506	0.7924	0.0458	0.8670	0.8787	0.7385	0.0574	0.8263	0.8533
BMP [43]	0.8682	0.0447	0.9108	0.9137	0.8705	0.0389	0.9065	0.9373	0.7578	0.0753	0.8431	0.8420	0.7453	0.0490	0.8616	0.8599	0.6917	0.0636	0.8093	0.8375
LFR [80]	0.8799	0.0525	0.8968	0.9005	0.8751	0.0396	0.9046	0.9313	0.7613	0.1066	0.8045	0.7992	0.7030	0.0834	0.8089	0.8059	0.6656	0.1030	0.7800	0.7799
Amulet [41]	0.8683	0.0589	0.8941	0.9011	0.8426	0.0501	0.8860	0.9122	0.7574	0.0997	0.8183	0.8016	0.6775	0.0846	0.8039	0.7939	0.6472	0.0976	0.7805	0.7787
UCF [81]	0.8439	0.0690	0.8834	0.8923	0.8233	0.0612	0.8742	0.9027	0.7261	0.1155	0.8055	0.8042	0.6307	0.1122	0.7823	0.7625	0.6206	0.1204	0.7599	0.7647
DLS [21]	0.8219	0.0860	0.8064	0.8655	0.8081	0.0696	0.7986	0.8788	0.7071	0.1301	0.7234	0.7925	-	-	-	-	0.6453	0.0895	0.7249	0.8016
ELE [22]	0.7545	0.1201	0.7426	0.8201	0.7053	0.1118	0.7127	0.8097	0.6444	0.1614	0.6682	0.7432	0.5765	0.1272	0.6704	0.7479	0.5752	0.1215	0.6763	0.7502
ELD [82]	0.8169	0.0783	0.8413	0.8835	0.7764	0.0719	0.8230	0.8845	0.7138	0.1206	0.7605	0.8055	0.6246	0.0924	0.7534	0.7844	0.6141	0.0909	0.7513	0.7771
MDF [17]	0.8068	0.1050	0.7761	0.8462	0.7843	0.1292	0.8101	0.8708	0.7020	0.1420	0.6959	0.7596	0.6692	0.0935	0.7330	0.8091	0.6443	0.0916	0.7208	0.7997

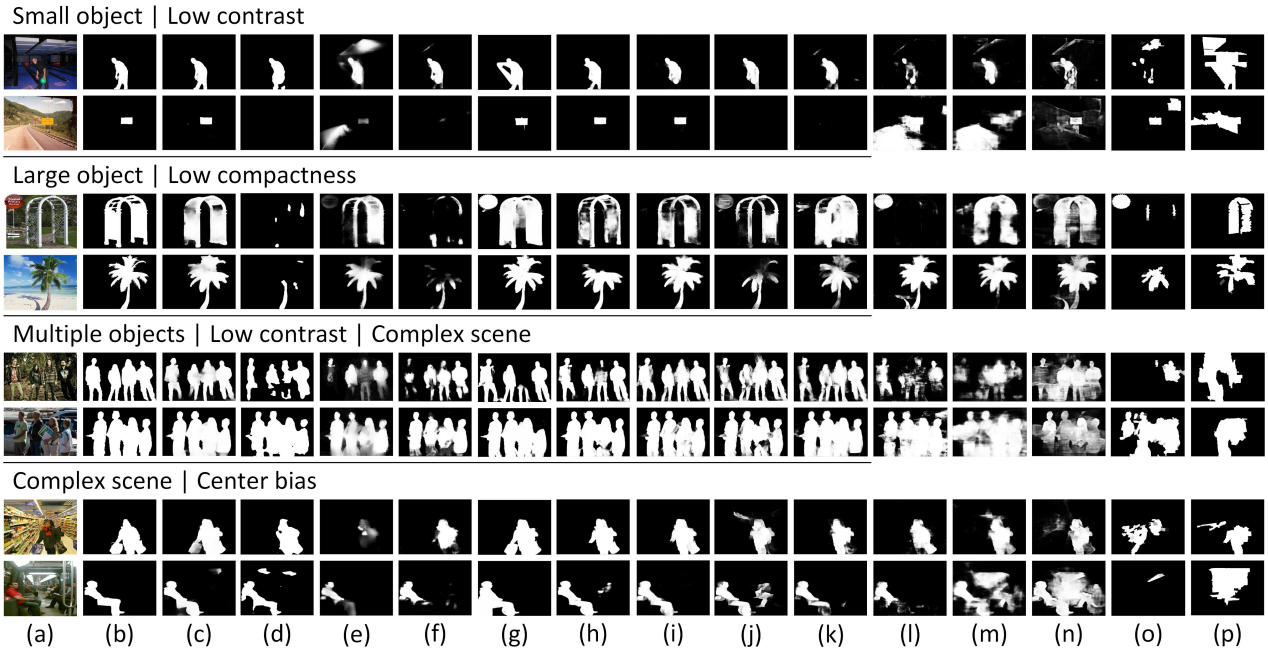


Fig. 22. Detection results of different methods. (a) Image; (b) GT; (c) OURS; (d) TSPOANet [26]; (e) JointCRF [76]; (f) ToHR [44]; (g) BASNet [77]; (h) CPD [78]; (i) PoolNet [45]; (j) AFNet [79]; (k) BMP [43]; (l) LFR [80]; (m) Amulet [41]; (n) UCF [81]; (o) DLS [21]; (p) ELE [22].

TABLE 4
Running time of some methods. It can be found that our method is much faster than CapsNet [25], JointCRF [76], ToHR [44], and MDF [17]. Especially, compared with the original CapsNet, our method can greatly accelerate the running speed by 65%, which benefits from the use of the proposed two-stream strategy.

	TSPORTNet	CapsNet [25]	JointCRF [76]	ToHR [44]	BMP [43]	LFR [80]	Amulet [41]	UCF [81]	MDF [17]
Time (second)	0.35	0.54	0.48	0.39	0.05	0.09	0.07	0.15	8
Crop size	352×352	352×352	240×320	384×384	256×256	384×384	256×256	448×448	400×300

6 Discussion and Future Work

6.1 Potential of Part-Object Relational Property

As the part-object relationships are essential for the object, the spirit, *i.e.*, detecting part-object relationships, presented in this work can help improve the performance of various vision tasks, especially object-centric tasks. To this end, we have extended our work to semantic segmentation and lesion detection. The preliminary results reveal: 1) For the segmentation task, *i.e.*, the pixel-level recognition task, by detecting associated object parts, our algorithm can help to recognize all pixels belonging to a certain object and thus segment the whole object out from complicated backgrounds; 2) For the object detection task, part-object relationships can help to find the complete object region, rather than discriminative parts only. It enables us to recognize the right interest proposals for accurate classification and localization.

To provide some context, we elaborate on how we expand our spirit to segment different objects in our daily-life images (*i.e.*, conducting semantic segmentation) and to detect the lesion regions in medical imaging data (*i.e.*, conducting universal lesion detection) in the supplementary materials.

6.2 Future Work

In the future, we will focus on several urgent issues. First, we will expand the property of part-object relationships to wide applications, *e.g.*, segmentation and object detection, by finding the intrinsic relations between the part-object relational property and these tasks.

Secondly, high computational complexity and a large number of parameters are the general problems faced by CapsNet related works. Therefore, a lightweight CapsNet is in demand for CapsNet to be used for wide applications. In the future, we will pay attention to this problem with two potential solutions: 1) Making the adjacent-layer capsule connections sparse can reduce the parameters by a large margin; 2) Designing a more intelligent capsule routing algorithm to trade off the accuracy of capsule assignments and computation speed is worth investigating.

7 Conclusion

In this paper, we have investigated a part-object relational visual saliency for the purpose of remedying the problem of incomplete segmentation of the salient object in a scene, which is accomplished by involving a new salient property of part-object relationships provided by CapsNet in salient object detection. To achieve this, we have presented a deep TSPORTNet, where a two-stream strategy is adopted to implement CapsNet, helping to reduce the network computation budgets while diminishing some noisy capsule assignments. A correlation-aware capsule routing algorithm is also presented by preserving the preceding capsule correlations between different capsule types across adjacent layers in favor of more accurate capsule assignments. With the part-object relationships discovered, TSPORTNet outputs a capsule wholeness map, which further aggregates with multi-level features to achieve the final saliency map. Extensive evaluations have verified the effectiveness and superiority of the proposed salient object detector. However, the network parameters and computation complexity of CapsNet are still obstacles for the large-scale dense segmentation and detection. In the future, we will investigate the issue of making CapsNet a light tool for large-scale dense segmentation and detection tasks.

Acknowledgment

This work is supported by the National Natural Science Foundation of China under Grant No. 62001341 and 61773301.

References

[1] J. Han, K. N. Ngan, M. Li, and H.-J. Zhang, "Unsupervised extraction of visual attention objects in color images," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 1, pp. 141–145, 2006.

[2] S. J. Oh, R. Benenson, A. Khoreva, Z. Akata, M. Fritz, B. Schiele *et al.*, "Exploiting saliency for object segmentation from image level labels," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4410–4419.

[3] J. Han, E. J. Pauwels, and P. De Zeeuw, "Fast saliency-aware multi-modality image fusion," *Neurocomputing*, vol. 111, pp. 70–80, 2013.

[4] U. Rutishauser, D. Walther, C. Koch, and P. Perona, "Is bottom-up attention useful for object recognition?" in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2004, pp. 37–44.

[5] Y. Wei, X. Liang, Y. Chen, X. Shen, M.-M. Cheng, J. Feng, Y. Zhao, and S. Yan, "Stc: A simple to complex framework for weakly-supervised semantic segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 11, pp. 2314–2320, 2017.

[6] C. Guo and L. Zhang, "A novel multiresolution spatiotemporal saliency detection model and its applications in image and video compression." *IEEE Transactions on Image Processing*, vol. 19, no. 1, pp. 185–198, 2010.

[7] L. Itti, "Automatic foveation for video compression using a neurobiological model of visual attention," *IEEE Transactions on Image Processing*, vol. 13, no. 10, pp. 1304–1318, 2004.

[8] J. Guo, T. Ren, L. Huang, X. Liu, M.-M. Cheng, and G. Wu, "Video salient object detection via cross-frame cellular automata," in *Proceedings of the IEEE International Conference on Multimedia and Expo*, 2017, pp. 325–330.

[9] Y. Gao, M. Wang, D. Tao, R. Ji, and Q. Dai, "3-d object retrieval and recognition with hypergraph analysis," *IEEE Transactions on Image Processing*, vol. 21, no. 9, pp. 4290–4303, 2012.

[10] M.-M. Cheng, Q.-B. Hou, S.-H. Zhang, and P. L. Rosin, "Intelligent visual media processing: When graphics meets vision," *Journal of Computer Science and Technology*, vol. 32, no. 1, pp. 110–121, 2017.

[11] R. Zhao, W. Ouyang, and X. Wang, "Unsupervised salience learning for person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 3586–3593.

[12] X. Li, H. Lu, L. Zhang, X. Ruan, and M.-H. Yang, "Saliency detection via dense and sparse reconstruction," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 2976–2983.

[13] C. Yang, L. Zhang, H. Lu, X. Ruan, and M.-H. Yang, "Saliency detection via graph-based manifold ranking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 3166–3173.

[14] W.-C. Tu, S. He, Q. Yang, and S.-Y. Chien, "Real-time salient object detection with a minimum spanning tree," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2334–2342.

[15] N. Li, B. Sun, and J. Yu, "A weighted sparse coding framework for saliency detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5216–5223.

[16] A. Borji, M.-M. Cheng, H. Jiang, and J. Li, "Salient object detection: A benchmark," *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 5706–5722, 2015.

[17] G. Li and Y. Yu, "Visual saliency based on multiscale deep features," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5455–5463.

[18] N. Liu and J. Han, "Dhsnet: Deep hierarchical saliency network for salient object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 678–686.

[19] Z. Luo, A. K. Mishra, A. Achkar, J. A. Eichel, S. Li, and P.-M. Jodoin, "Non-local deep features for salient object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6609–6617.

[20] Y. Liu, J. Han, Q. Zhang, and C. Shan, "Deep salient object detection with contextual information guidance," *IEEE Transactions on Image Processing, DOI: 10.1109/TIP.2019.2930906*, 2019.

[21] P. Hu, B. Shuai, J. Liu, and G. Wang, "Deep level sets for salient object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2300–2309.

[22] C. Xia, J. Li, X. Chen, A. Zheng, and Y. Zhang, "What is and what is not a salient object? learning salient object detector by ensembling linear exemplar regressors," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4321–4329.

[23] G. E. Hinton, A. Krizhevsky, and S. D. Wang, "Transforming auto-encoders," in *Proceedings of the International Conference on Artificial Neural Networks*, 2011, pp. 44–51.

[24] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Advances in Neural Information Processing Systems*, 2017, pp. 3856–3866.

[25] G. E. Hinton, S. Sabour, and N. Frosst, "Matrix capsules with em routing," in *Proceedings of the International Conference on Learning Representations*, 2018, pp. 3856–3866.

[26] Y. Liu, Q. Zhang, D. Zhang, and J. Han, "Employing deep part-object relationships for salient object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 1232–1241.

[27] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 11, pp. 1254–1259, 1998.

[28] D. A. Klein and S. Frintrop, "Center-surround divergence of feature statistics for salient object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2011, pp. 2214–2219.

[29] M.-M. Cheng, N. J. Mitra, X. Huang, P. H. Torr, and S.-M. Hu, "Global contrast based salient region detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 569–582, 2015.

[30] T. Liu, Z. Yuan, J. Sun, J. Wang, N. Zheng, X. Tang, and H.-Y. Shum, "Learning to detect a salient object," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 2, pp. 353–367, 2011.

[31] H. Lu, X. Li, L. Zhang, X. Ruan, and M.-H. Yang, "Dense and sparse reconstruction error based saliency descriptor," *IEEE Transactions on Image Processing*, vol. 25, no. 4, pp. 1592–1603, 2016.

[32] H. Peng, B. Li, H. Ling, W. Hu, W. Xiong, and S. J. Maybank, "Salient object detection via structured matrix decomposition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 818–832, 2017.

[33] J. Kim, D. Han, Y.-W. Tai, and J. Kim, "Salient region detection via high-dimensional color transform and local spatial support," *IEEE Transactions on Image Processing*, vol. 25, no. 1, pp. 9–23, 2016.

[34] R. Cong, J. Lei, H. Fu, Q. Huang, X. Cao, and C. Hou, "Co-saliency detection for rgbd images based on multi-constraint feature matching and cross label propagation," *IEEE Transactions on Image Processing*, vol. 27, no. 2, pp. 568–579, 2018.

[35] H. Fu, X. Cao, and Z. Tu, "Cluster-based co-saliency detection," *IEEE Transactions on Image Processing*, vol. 22, no. 10, pp. 3766–3778, 2013.

[36] X. Cao, Z. Tao, B. Zhang, H. Fu, and W. Feng, "Self-adaptively weighted co-saliency detection via rank constraint," *IEEE Transactions on Image Processing*, vol. 23, no. 9, pp. 4175–4186, 2014.

[37] Y. Liu, J. Han, Q. Zhang, and L. Wang, "Salient object detection via two-stage graphs," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 4, pp. 1023–1037, 2018.

[38] Y. Liu, Q. Zhang, J. Han, and L. Wang, "Salient object detection employing robust sparse representation and local consistency," *Image and Vision Computing*, vol. 69, pp. 155–167, 2018.

[39] R. Zhao, W. Ouyang, H. Li, and X. Wang, "Saliency detection by multi-context deep learning," in *Proceedings*

of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1265–1274.

[40] L. Wang, H. Lu, X. Ruan, and M.-H. Yang, "Deep networks for saliency detection via local estimation and global search," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3183–3192.

[41] P. Zhang, D. Wang, H. Lu, H. Wang, and X. Ruan, "Amulet: Aggregating multi-level convolutional features for salient object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 202–211.

[42] N. Liu, J. Han, and M.-H. Yang, "Picanet: Learning pixel-wise contextual attention for saliency detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3089–3098.

[43] L. Zhang, J. Dai, H. Lu, Y. He, and G. Wang, "A bi-directional message passing model for salient object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1741–1750.

[44] Y. Zeng, P. Zhang, J. Zhang, Z. Lin, and H. Lu, "Towards high-resolution salient object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 7234–7243.

[45] J.-J. Liu, Q. Hou, M.-M. Cheng, J. Feng, and J. Jiang, "A simple pooling-based design for real-time salient object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3917–3926.

[46] W. Zhao, J. Ye, M. Yang, Z. Lei, S. Zhang, and Z. Zhao, "Investigating capsule networks with dynamic routing for text classification," *arXiv preprint arXiv:1804.00538*, 2018.

[47] K. Duarte, Y. Rawat, and M. Shah, "Videocapsulenet: A simplified network for action detection," in *Advances in Neural Information Processing Systems*, 2018, pp. 7610–7619.

[48] I. Biederman, "Recognition-by-components: a theory of human image understanding." *Psychological review*, vol. 94, no. 2, pp. 115–148, 1987.

[49] M. Jüttner, T. Caelli, and I. Rentschler, "Recognition-by-parts: a computational approach to human learning and generalization of shapes," *Biological cybernetics*, vol. 74, no. 6, pp. 521–535, 1996.

[50] F. Solina and R. Bajcsy, "Recovery of parametric models from range images: The case for superquadrics with global deformations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 2, pp. 131–147, 1990.

[51] A. Jaklic, A. Leonardis, and F. Solina, *Segmentation and recovery of superquadrics*. Springer Science & Business Media, 2013, vol. 20.

[52] J. Krivic and F. Solina, "Part-level object recognition using superquadrics," *Computer Vision and Image Understanding*, vol. 95, no. 1, pp. 105–126, 2004.

[53] A. Jaklic, "Construction of cad models from range images," *Ph. D's thesis, University of Ljubljana*, 1997.

[54] A. P. Pentland, "Automatic extraction of deformable part models," *International Journal of Computer Vision*, vol. 4, no. 2, pp. 107–126, 1990.

[55] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester, "Cascade object detection with deformable part models," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2010, pp. 2241–2248.

[56] R. Girshick, F. Iandola, T. Darrell, and J. Malik, "Deformable part models are convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 437–446.

[57] Y. Zhao, J. Li, Y. Zhang, Y. Song, and Y. Tian, "Ordinal multi-task part segmentation with recurrent prior generation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[58] P. Wang, X. Shen, Z. Lin, S. Cohen, B. Price, and A. L. Yuille, "Joint object and part segmentation using deep learned potentials," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1573–1581.

[59] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.

[60] L. Chen, H. Zhang, J. Xiao, L. Nie, J. Shao, W. Liu, and T.-S. Chua, "Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5659–5667.

[61] X. Chu, W. Yang, W. Ouyang, C. Ma, A. L. Yuille, and X. Wang, "Multi-context attention for human pose estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1831–1840.

[62] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, "Residual attention network for image classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3156–3164.

[63] L. Zhang, J. Wu, T. Wang, A. Borji, G. Wei, and H. Lu, "A multistage refinement network for salient object detection," *IEEE Transactions on Image Processing*, vol. 29, pp. 3534–3545, 2020.

[64] T. Wang, L. Zhang, S. Wang, H. Lu, G. Yang, X. Ruan, and A. Borji, "Detect globally, refine locally: A novel approach to saliency detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3127–3135.

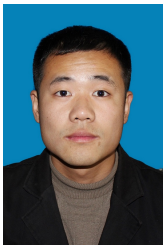
[65] Q. Yan, L. Xu, J. Shi, and J. Jia, "Hierarchical saliency detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1155–1162.

[66] Y. Li, X. Hou, C. Koch, J. M. Rehg, and A. L. Yuille, "The secrets of salient object segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 280–287.

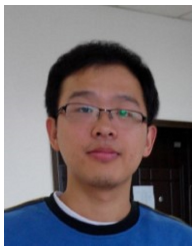
[67] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.

[68] L. Wang, H. Lu, Y. Wang, M. Feng, D. Wang, B. Yin, and X. Ruan, "Learning to detect salient objects with image-level supervision," in *Proccedings of the IEEE Conference Computer Vision and Pattern Recognition*, 2017, pp. 136–145.

[69] R. Achanta, S. Hemami, F. Estrada, and S. Susstrunk, "Frequency-tuned salient region detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 1597–1604.
[70] D.-P. Fan, M.-M. Cheng, Y. Liu, T. Li, and A. Borji, "Structure-measure: A new way to evaluate foreground maps," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 4548–4557.
[71] D.-P. Fan, C. Gong, Y. Cao, B. Ren, M.-M. Cheng, and A. Borji, "Enhanced-alignment measure for binary foreground map evaluation," in *Proceedings of the International Joint Conference on Artificial Intelligence*, 2018, pp. 698–704.
[72] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: a system for large-scale machine learning," in *Operating System Design and Implementation*, 2016, pp. 265–283.
[73] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proceedings of the International Conference on Learning Representation*, 2015.
[74] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[75] M. M. Cheng, G. X. Zhang, N. J. Mitra, X. Huang, and S. M. Hu, "Global contrast based salient region detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 409–416.
[76] Y. Xu, D. Xu, X. Hong, W. Ouyang, R. Ji, M. Xu, and G. Zhao, "Structured modeling of joint deep feature and prediction refinement for salient object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 3789–3798.
[77] X. Qin, Z. Zhang, C. Huang, C. Gao, M. Dehghan, and M. Jagersand, "Basnet: Boundary-aware salient object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7479–7489.
[78] Z. Wu, L. Su, and Q. Huang, "Cascaded partial decoder for fast and accurate salient object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3907–3916.
[79] M. Feng, H. Lu, and E. Ding, "Attentive feedback network for boundary-aware salient object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1623–1632.
[80] P. Zhang, W. Liu, H. Lu, and C. Shen, "Salient object detection by lossless feature reflection," in *Proccedings of the International Joint Conference on Artificial Intelligence*, 2018.
[81] P. Zhang, D. Wang, H. Lu, H. Wang, and B. Yin, "Learning uncertain convolutional features for accurate saliency detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 212–221.
[82] G. Lee, Y.-W. Tai, and J. Kim, "Eld-net: An efficient deep learning architecture for accurate saliency detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 7, pp. 1599–1610, 2018.



Yi Liu received the B. S. degree from Nanjing Institute of Technology, China, in 2012, the M. S. degree from the Dalian University, China, in 2015, and the Ph. D. degree from Xidian University, China, in 2019. From 2018 to 2019, he was a visiting scholar at Lancaster University. He is currently working as a postdoctoral researcher at Xidian University. His research interests include computer vision and deep learning.



Dingwen Zhang Dingwen Zhang received the BE and PhD degrees from the Northwestern Polytechnical University, Xian, China, in 2012 and 2018, respectively. From 2015 to 2017, he was a visiting scholar at Carnegie Mellon University, Pittsburgh, USA. He is currently an associate professor with Xidian University. His research interests include computer vision and multimedia processing, especially on saliency detection, cosaliency detection, and weakly supervised learning.

Qiang Zhang received the B.S. degree in automatic control, the M.S. degree in pattern recognition and intelligent systems, and the Ph.D. degree in circuit and system from Xidian University, China, in 2001,2004, and 2008, respectively. He was a Visiting Scholar with the Center for Intelligent Machines, McGill University, Canada. He is currently a professor with the Automatic Control Department, Xidian University, China. His current research interests include image processing and pattern recognition.

Jungong Han is currently a Chair Professor with the Department of Computer Science, Aberystwyth University, U.K. He also holds an honorary professorship with the University of Warwick, U.K. His research interests include computer vision, artificial intelligence and machine learning. He has published over 50 IEEE/ACM Transactions papers and 40 A* conference papers.